CVS @ OpenHard 用户说明书

Xup China

Walkie

(xiashengwalk@gmail.com)

注意：

本文档中所使用的 cvs 的数字用户，比如这里举例的 143，以及还有 142～200 等用户，出于安全性的考虑，已经将数字名字修改成 x143，或者 x142~x200 这样的名字，密码未改变。所以在使用本文档的时候，这里所指的 143 用户就是 x143，143 文件夹就是 x143，特别是图片中的 143。均需要改成 x143。给各位用户带来不便，还请见谅。

2007-11-01 Version 1.2

First Open

2007-12-01 Version 1.3

扩充 3.6 小节关于如何使用 wincvs 添加非文本文件，如 word，ppt，rar 等。

2007-12-09 Version1.4

1、将 OpenHard 服务器上的所有纯数字 cvs 用户修改成非数字，均改成 x+number 的样子。并且将服务器上的用户对应的工程目录页改成 x+number 的样子。增加安全性的同时，方便大家平时的使用。

2、开放了 cvsweb 的使用。

3、说明如何使用 wincvs 进行初次 checkout 工程目录，详细请见 3.4 小节。

目录

1 CVS 概述

1.1 CVS 是什么？

CVS 是一个版本控制系统。 使用它，可以记录下你的源文件的历史。

例如，修改软件时可能会不知不觉混进一些 bug，而且可能过了很久你才会察觉到它们的存在。有了 CVS，你可以很容易地恢复旧版本，并从中看出到底是哪个修改导致了这个 bug。 有时这是很有用的。

你当然可以把曾经创建的每个文件的所有版本都保存下来。 但这会浪费大量的磁盘空间。 而 CVS 用一种聪明的办法把一个文件的所有版本保存在一个文件里，仅仅保存不同版本之间的差异。

如果你是项目开发组的一员，CVS 也会帮助你。除非极为小心，成员之间很容易互相覆盖文件。 一些编辑器，如 gnu Emacs，会保证同一时间内同一文件绝不会被两个人修改。不幸的是，如果有人用了另外的编辑器，这种保护就没用了。CVS 用隔离开不同的开发者解决了这个问题。每个开发者在他自己的目录里工作，等每一个开发者都完成了他们自己的工作后，CVS 会将它们合并到一起。

CVS 最初由 Dick Grune 在 1986 年 12 月以 shell 脚本的形式发布在 comp.sources.unix 的新闻组第 6 卷里。虽然当前的 CVS 中没什么代码来自于这些 shell 脚本，但许多 CVS 的冲突解决算法是从它们来的。1989 年 4 月，Brian Berliner 设计了 CVS 并编写了代码。之后 Jeff Polk 帮助 Brian 设计了 CVS 模块和销售商分支支持。

你可以从好多渠道得到 CVS，包括从因特网上自由下载。有关下载 CVS 和其它与 CVS 相关的话题，请访问下面网址：

http://www.CVShome.org/

1.2 CVS @ OpenHard

为了方便众多教育网的用户，OpenHard 的 CVS 服务器挂在教育网内，CVS 的具体的时候用法见 2. CVS 的基本用法，接下来将叙述 OpenHard 的 CVS 仓库中用户和文件的权限问题。

各个小组的 CVS 用户名是各小组注册邮箱的用户名经过一些处理之后得到的，密码是一个随机的密码，到时用户名和密码将通过邮件方式发送到各位的邮箱。

这个仓库的目录结构是这样的：

$CVSROOT /
mhuasong
ybs
zhouyue
hongyuanquan
zhaotao
weibit
gongjr
wang
mengxy
lshc
wangweibit
zhaoming
stonestrong

zhys
jianyuwang
wxu
zhujianping
liushuyong
zhaoshixia2
liangqing
142
143
144
145
146
147
148
149
150

$CVSROOT 是仓库的根目录，目录下为各个小组都创建了一个跟用户名同名的，只属于该小组的工程目录。

在权限方面，做了一些设置，各小组在自己的工程目录下面有读和写的权限，可以下载文件，或者文件夹（这是读权限），也可以上传文件，文件夹（这是写权限），但是对于根目录下面的其他小组的文件夹以及他们下面的子目录只有读的权限。

例如用户 143，那么它在 CVS 的仓库中的工程目录是 143，它所拥有的权限就是对于 143 这个目录以及子目录的读和写的权限，但是对于 143 之外的所有文件和文件夹都是只有读取的权限，没有写的权限，那么他就不可以在除了 143 文件夹之外的文件夹（包括根目录）下面上传文件或者修改他人的文件。

2 CVS 的基本用法（Linux）

以下将以用户 143 作为例子来演示如何使用 cvs 进行工作。

2.1 CVS 登录

使用如下方式登录：

```
[walkie@localhost ~]$ export CVSROOT=:pserver:143@124.128.158.44:/var/cvs
[walkie@localhost ~]$ cvs login
Logging in to :pserver:143@124.128.158.44:2401/var/cvs
CVS password:
```

会要求输入密码，输入密码后按回车，如果没有任何错误信息，则表示登录成功。如果是第一次登录，需要将远程服务器上的工程目录下载到本地，本地的目录称为沙箱。下面将介绍如何创建沙箱。

这里可是使用的 IP 地址是 124128.158.44，这个有时会很慢。教育网的用户也可以使用以下这个 IP，速度会快一点：202.194.64.61

2.2 创建沙箱（远程仓库的一个本地备份）

沙箱其实只是远程服务器上的一个本地备份，每次编程人员在本地沙箱中修该文件，在一天工作结束或者工作暂停，需要做一个备份时，可以将本地的沙箱中的做的新的修该导入远程仓库中。下面将说明如何使用 checkout 命令创建沙箱：

```
[walkie@localhost ~]$ cvs checkout 143
cvs checkout: Updating 143
[walkie@localhost ~]$ ls
143  Desktop
```

checkout 命令能使用模块名，或者是一个相对于 $CVSROOT 的路径，如同上面的例子。或者也可以使用如下方式

```
[Walkie@localhost ~]$ cvs checkout 143/module1
```

其中 module1 是一个 143 目录下面的子文件夹。

2.3 添加文件

本地的沙箱创建好之后，就可以进行进行项目开放了。这里 2.3.1 介绍如何添加个别文件，2.3.2 介绍如何导入大量文件。下图是一个概述：

```
walkie@localhost:~/143 - Shell - Konsole
Session  Edit  View  Bookmarks  Settings  Help
[walkie@localhost ~]$ rm -rf 14*
[walkie@localhost ~]$ ls                    登录并输入密码
Desktop
[walkie@localhost ~]$ export CVSROOT=:pserver:143@124.128.158.44:/var/cvs
[walkie@localhost ~]$ cvs login
Logging in to :pserver:143@124.128.158.44:2401/var/cvs
CVS password:
[walkie@localhost ~]$ cvs checkout 143
cvs checkout: Updating 143
[walkie@localhost ~]$ ls
143  Desktop
[walkie@localhost ~]$ cd 143/
[walkie@localhost 143]$ ls
CVS
[walkie@localhost 143]$ touch Welcome.c
[walkie@localhost 143]$ mkdir module1
[walkie@localhost 143]$ touch submudole.c
[walkie@localhost 143]$ mv submudole.c module1/
[walkie@localhost 143]$ ls
CVS  module1  Welcome.c
[walkie@localhost 143]$ cvs add module1 module1/*.c Welcome.c
? module1/submudole.c
Directory /var/cvs/143/module1 added to the repository
cvs add: nothing known about module1/submudole.c
cvs add: scheduling file `Welcome.c' for addition
cvs add: use 'cvs commit' to add this file permanently
[walkie@localhost 143]$ cvs commit
cvs commit: Examining .
cvs commit: Examining module1
? module1/submudole.c
RCS file: /var/cvs/143/Welcome.c,v
done
Checking in Welcome.c;
/var/cvs/143/Welcome.c,v  <--  Welcome.c
initial revision: 1.1
done
[walkie@localhost 143]$
Shell   2 Shell
walkie@localhost:  root@edu:/home/wa  xinetd.d - Konc
walkie@localhost:/h  (I)Cooka@MicroE  How to disable
```

2.3.1 添加个别文件

使用命令 cvs add <file>命令，可以添加个别文件，如下图所示。这里首先创建了一个 Welcome.c 的 source 文件，创建了一个 module1 的文件夹，并在这个文件夹下面创建了一个 submodule.c 的 source 文件。然后添加。使用 add 命令文件，只有在 cvs commit（这个命令的具体说明将在下面章节叙述）时才会真正的把文件上传到远程沙箱中，使用这个命令会跳出一个对话框让你输入修该日志（见 2.4）。如下图：

```
[walkie@localhost ~]$ cd 143/
[walkie@localhost 143]$ ls
CVS
[walkie@localhost 143]$ touch Welcome.c
[walkie@localhost 143]$ mkdir module1
[walkie@localhost 143]$ touch submudole.c
[walkie@localhost 143]$ mv submudole.c module1/
[walkie@localhost 143]$ ls
CVS  module1  Welcome.c
[walkie@localhost 143]$ cvs add module1 module1/*.c Welcome.c
? module1/submudole.c
Directory /var/cvs/143/module1 added to the repository
cvs add: nothing known about module1/submudole.c
cvs add: scheduling file `Welcome.c' for addition
cvs add: use 'cvs commit' to add this file permanently
[walkie@localhost 143]$ cvs commit
cvs commit: Examining .
cvs commit: Examining module1
? module1/submudole.c
RCS file: /var/cvs/143/Welcome.c,v
done
Checking in Welcome.c;
/var/cvs/143/Welcome.c,v  <--  Welcome.c
initial revision: 1.1
done
[walkie@localhost 143]$
```

注：这里添加子目录下面的 submodule.c 时，没有识别出来，也许是出于偷懒的心理。使用如下命令可以补过。

```
[Walkie@localhost 143]$ cvs    add  module1/submodule.c
[Walkie@localhost 143]$ cvs    commit.
```

经过上述命令之后服务器上的情况如下图所示：

```
[root@edu walkie]# cd /var/cvs/143
[root@edu 143]# ls
module1  Welcome.c,v
[root@edu 143]# cd module1/
[root@edu module1]# ls
[root@edu module1]# ls
submudole.c,v
[root@edu module1]#
```

Shell

2.3.2 添加大量文件

当你开始使用 cvs 时，你可能已经有几个项目可以置于 cvs 控制下了。 在这种情况下，最容易的方法就是使用：导入 import 命令。通过一个例子是最容易解释如何使用它的。假定你现在想把沙箱 143 下面 module2 文件夹及其他内部的所有文件都导入 cvs 远程仓库中，你可以这样做：

```
[walkie@localhost 143]$ mkdir module2
[walkie@localhost 143]$ cd module2/
[walkie@localhost module2]$ touch a.c
[walkie@localhost module2]$ touch b.c
[walkie@localhost module2]$ touch c.c
[walkie@localhost module2]$ ls
a.c  b.c  c.c
[walkie@localhost module2]$ cvs import -m "test" 143/module2 v_0_1 start
N 143/module2/b.c
N 143/module2/a.c
N 143/module2/c.c

No conflicts created by this import
```

请注意，要在 module2 文件夹内部，使用 import 命令，其中 143/module2 文件夹是远程仓库中的文件夹名字，就是你想将本地文件放入的远程文件夹的名字。当然你也可以将其放入其他文件夹内，比如 143/module1，这样本地 module2 文件夹内的内部经过这次 import 之后，会把所有的文件都导入远程仓库中的 143/module1 文件夹内部。如果你要导入的文件夹在远程仓库中不存在，import 命令会自己创建一个。

如果你没有使用 `-m' 标志记录一个日志信息，cvs 将启动一个编辑器并且提示输入信息。 v_0_1 字符串是销售商标签（*vendor tag*），而 `start' 是发行标签（*release tag*）。 它们没有什么特别的意义，仅仅是因为 cvs 的需要才放在这里。 参阅 Tracking sources, 以得到更多的这方面信息。

2.3.3 添加非文本文件

如果你想导入的一些文件是二进制代码，你可以使用一些特殊的方法表明这些文件是否是二进制文件。这里注意使用 doc 和 pdf 之类的文件虽然也是文档文件，但是不是文本文件，所以要当成二进制代码文件来处理，要不，cvs 会在这里文件的一些地方插入一些标识符，破坏文件。这里要介绍一个 cvs 的特性，封装(Wrappers) 它可以让你控制基于被操作文件的文件名 的设定。设定中 `-k' 用于二进制文件，`-m' 用于不可合并的文本文件。

如下图所示：

```
[walkie@localhost 143]$ ls -la
total 112
drwxrwxr-x  4 walkie walkie  4096 Oct 27 18:51 .
drwx------ 21 walkie walkie  4096 Oct 27 18:50 ..
-rw-rw-r--  1 walkie walkie     0 Oct 27 18:04 a.c
drwxrwxr-x  2 walkie walkie  4096 Oct 27 18:41 CVS
drwxrwxr-x  3 walkie walkie  4096 Oct 27 17:48 module1
-rwxr-xr-x  1 walkie walkie 84696 Oct 27 18:51 test.pdf
-rw-rw-r--  1 walkie walkie     0 Oct 27 18:23 Welcome.c
-rw-rw-r--  1 walkie walkie    55 Oct 27 18:00 .#Welcome.c.1.2
[walkie@localhost 143]$ cvs add -kb test.pdf
cvs add: scheduling file `test.pdf' for addition
cvs add: use 'cvs commit' to add this file permanently
[walkie@localhost 143]$ cvs commit
cvs commit: Examining .
cvs commit: Examining module1
RCS file: /var/cvs/143/test.pdf,v
done
Checking in test.pdf;
/var/cvs/143/test.pdf,v  <--  test.pdf
initial revision: 1.1
done
```

2.4 修该并将变更提交给远程仓库

2.4.1 提交修该

一旦文件被放入本地沙箱之后，你就能一如往常般的进行编辑。文件经过编辑之后，除非进行提交的操作，否则变更之处并不会反映到仓库中。cvs commit 命令可以将沙箱中的变更之处上传到仓库。找到仓库中哪些文件需要变更之后，cvs commit 就会启动编辑器，让你输入日志信息。

例如下面的流程是我修该了 Welcome.c 文件并将变更上传到远程仓库，

总的流程：

```
initial revision: 1.1
done
[walkie@localhost 143]$ ls
CVS  module1  Welcome.c
[walkie@localhost 143]$ vim Welcome.c
[walkie@localhost 143]$ vim Welcome.c
[walkie@localhost 143]$ ls
CVS  module1  Welcome.c
[walkie@localhost 143]$ cvs commit
cvs commit: Examining .
cvs commit: Examining module1
Checking in Welcome.c;
/var/cvs/143/Welcome.c,v  <--  Welcome.c
new revision: 1.2; previous revision: 1.1
done
[walkie@localhost 143]$
```

编辑Welcome.c文件，在里面输入一些文字

版本号从原来的1.1变到1.2

Shell 2 Shell

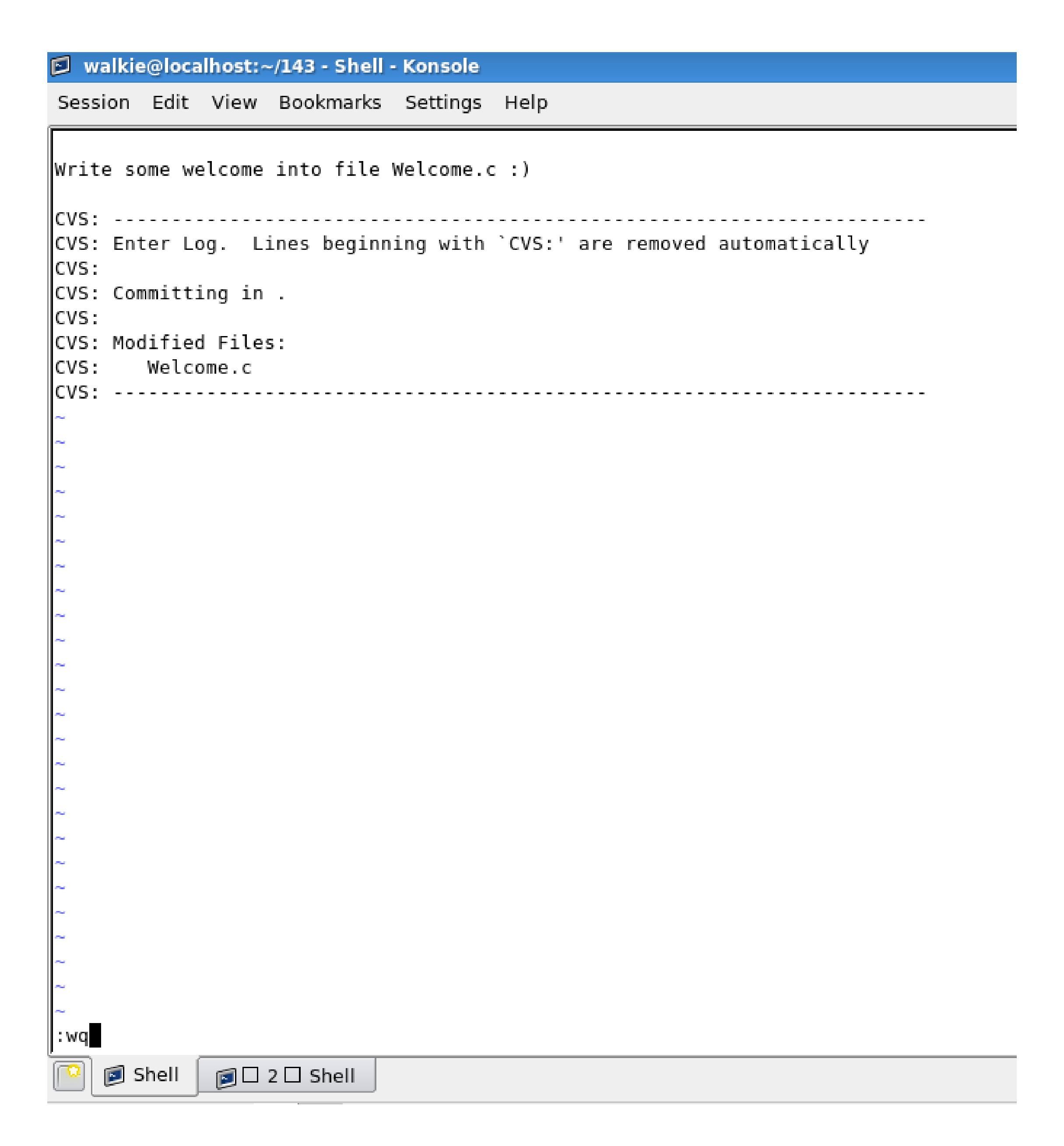
Vim 在编辑 Welcome.c 文件：

walkie@localhost:~/143 - Shell - Konsole

Session Edit View Bookmarks Settings Help

```
Hello everyone!
Welcome to OpenHard cvs!
Enjoy it.
~
~
~
~
~
~
~
~
```

Vim 在输入日志文件：



如果你不想在提交期间输入日志信息，可以使用如下的 cvs commit 命令：

[Walkie@localhost 143]$ cvs commit –m message

2.4.2 提交时机

如果你的变更不会影响其他人，就应该经常进行提交操作。只要你遇到不想重做的事，就要进行提交，你的操作成果就可以随时备份。如果是团队合作，时常提交和更新可以降低合并时碰上大困难的几率。

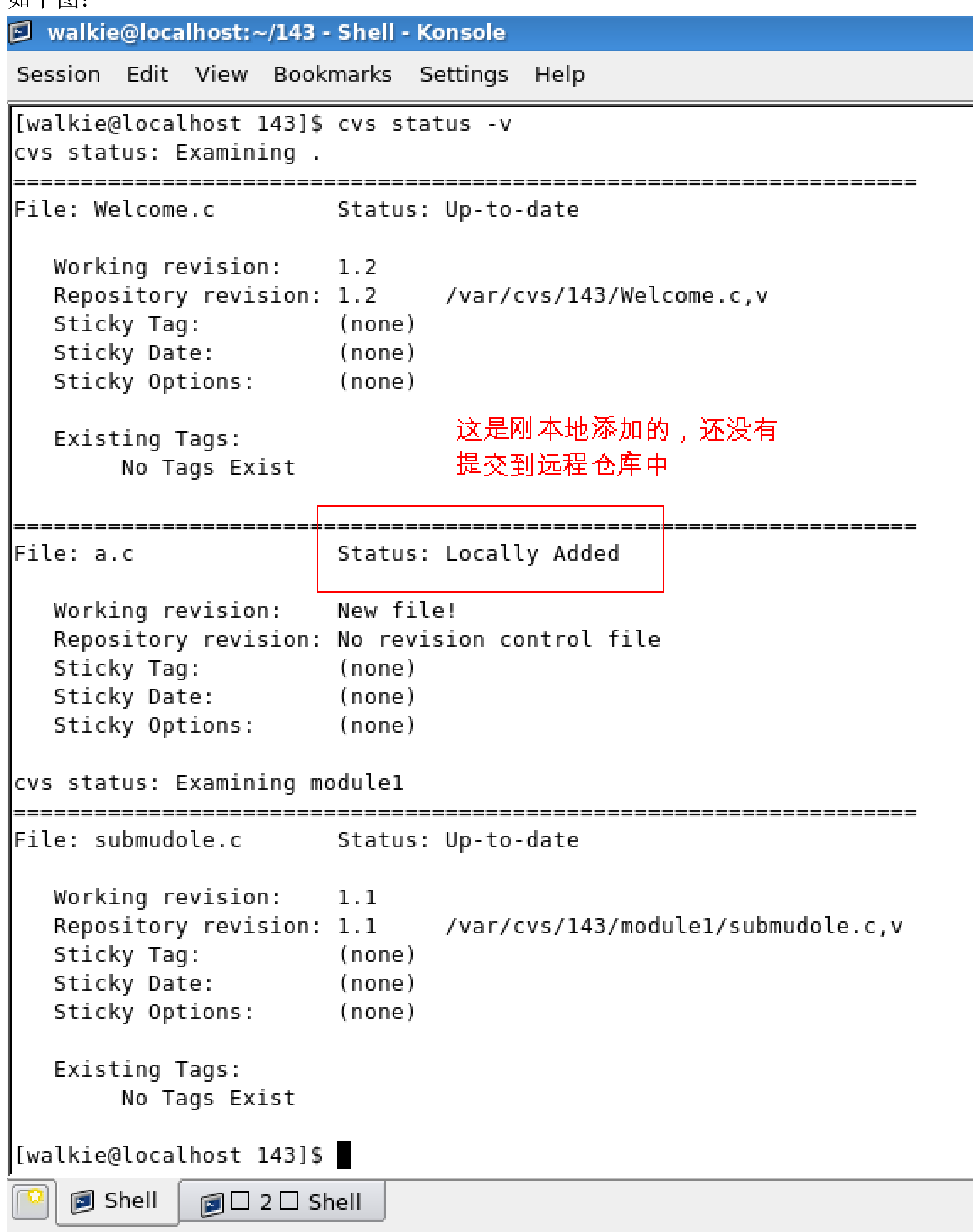
如果你的变更会影响其他人，尤其是会加大编译过程出错的风险，那么就应该在这种影响最小时进行提交。例如，试着在午餐前和下班前进行编译，如果可以编译成功，就进行提交。

你的团队应该决定使用何种提交机制，依据自己 cvs 的经验在多种机制中选择出适合

的方式。 如果提交文件太快，你可能甚至提交无法编译的文件。假如你的合作者更新了他的工作源文件而包括了你的错误文件，他也将无法编译该代码。而另一种情况，如果你很少提交，别人又无法得到你对代码所做的改进，并使冲突的几率增大。通常应该仅在确认可以编译后才可提交。有时要求通过测试工具的检查。

2.5 检查文件状态

cvs status 命令可让你很快得知哪些文件是最新的，以及哪些文件需要被提交或合并。如下图：



2.6 从远程仓库中更新本地沙箱的文件

2.6.1 获得更新

在前面的几节里讲了如何使用 cvs checkout 命令将本地沙箱中所做的修该变更上传到仓库中，而 cvs update 命令会下载远程仓库中的所有变更到沙箱。这两个命令彼此互补，你需要使用这两个命令以维持本地沙箱和远程仓库的同步。自从上一次提交之后，如你修该过本地沙箱中的文件，则使用 cvs commit；自从上一次更新之后，若你知道仓库又有了变更，则使用 cvs update。

```
[walkie@localhost 143]$ cvs update -d
cvs update: Updating .
A a.c
cvs update: Updating module1
[walkie@localhost 143]$ 
```



使用 CVS checkout 或 commit（提交），或 update 的时候，文件前面都会出现一个符号，这个符号很重要，错误信息组要从这个符号反映出来。这里你会在 cvs update 命令的输出中看到下列的符号：

A filename

此文件已经被新增到本地沙箱中，但是尚未被提交，例如这里的 a.c 文件

其他符号比如：

U 代表 update （表明你下载的是最新版本）

C 代表 conflict （表明你 commit 的时候 有问题）

M 代表 modify （表明你的文件已经被修改过）

2.6.2 更新时机

文件应该时常更新。工作之前一定要更新，这也许是早上和午餐之后的第一件工作。这样可以确保任何变更都会时常合并在一起。时常提交和更新可以降低合并时出大麻烦的几率。

在程序设计项目中，如果每个上传人员在提交之前都能确保程序可以编译，则他们的提交操作不会破坏你的程序。如果你的程序在更新之后无法运行，可以查出是谁做了变更而使得你的程序无法运行，然后立即和对方沟通。时常提交和更新可以提早解决这类问题。

2.7 取回某个文件的过去的修订版本

你可以使用 CVS 取回某个文件的过去的修订版本。

2.7.1 查找修订版本的标记（cvs log）

标记名称或者日志信息可以协助你确认哪个修订版有你想要的数据，例如在上例中，我创建了 Welcome.c 文件，并提交到远程仓库，版本号是 1.1，接下来我在本地沙箱中修该了一下这个文件，在里面添加了一些内容，再次提交之后，版本号是 1.2，那么 cvs log 的信息如下：

```
walkie@localhost:~/143 - Shell - Konsole
Session  Edit  View  Bookmarks  Settings  Help
cvs update: Updating module1
[walkie@localhost 143]$ cvs log
cvs log: Logging .

RCS file: /var/cvs/143/Welcome.c,v
Working file: Welcome.c
head: 1.2
branch:
locks: strict
access list:
symbolic names:
keyword substitution: kv
total revisions: 2;     selected revisions: 2
description:
----------------------------
revision 1.2
date: 2007/10/27 09:57:49;  author: 143;  state: Exp;  lines: +4 -0

Write some welcome into file Welcome.c :)
----------------------------
revision 1.1
date: 2007/10/27 09:45:47;  author: 143;  state: Exp;
This a test module :)
=============================================================================
cvs log: a.c has been added, but not committed
cvs log: Logging module1

RCS file: /var/cvs/143/module1/submudole.c,v
Working file: module1/submudole.c
head: 1.1
branch:
locks: strict
access list:
symbolic names:
keyword substitution: kv
total revisions: 1;     selected revisions: 1
description:
Shell   2 Shell
```

上图中，Welcome.c 有两个版本，一个是 1.1，当时的日记信息是“This is a test module ☺”另外一个是 1.2，当时的日记信息是“Write some welcome into file Welcome.c ☺”，这些日记就是前面，我们在提交时输入的日记信息。

同时 cvs log 还会显示其他的所有信息，比如这里它显示有一个文件（a.c）被添加了，但是还没有提交。以及另外一个文件没有什么改变。

2.7.2 还原到先前版本

比如下图：

```
================================================================================
[walkie@localhost 143]$ vim Welcome.c
[walkie@localhost 143]$ cvs commit
cvs commit: Examining .
cvs commit: Examining module1
cvs commit: Examining module2
Checking in Welcome.c;
/var/cvs/143/Welcome.c,v  <--  Welcome.c
new revision: 1.5; previous revision: 1.4
done
[walkie@localhost 143]$ vim Welcome.c
[walkie@localhost 143]$ ls
a.c  CVS  module1  module2  test.pdf  Welcome.c
[walkie@localhost 143]$ cvs commit
cvs commit: Examining .
cvs commit: Examining module1
cvs commit: Examining module2
Checking in Welcome.c;
/var/cvs/143/Welcome.c,v  <--  Welcome.c
new revision: 1.6; previous revision: 1.5
done
[walkie@localhost 143]$ cvs update -j 1.6 -j 1.5 Welcome.c
RCS file: /var/cvs/143/Welcome.c,v
retrieving revision 1.6
retrieving revision 1.5
Merging differences between 1.6 and 1.5 into Welcome.c
[walkie@localhost 143]$ cat Welcome.c
 Welcome to my CVS server!
[walkie@localhost 143]$
```

首先在 Welcome.c 写入了“Welcome to my CVS server!”，提交变成 revision1.5，然后再次修改 Welcome.c，加入语句“Test it yourself!”，提交，变成 revision1.6。

假设现在我们需要把当前的版本号恢复到 1.5，可是使用如下命令

[Walkie@localhost 143]$ cvs update –j 1.6 –j 1.5 Welcome.c

就可以恢复到版本 1.5 的内容了，但是这个时候 Welcome.c 变成了版本号 1.7，使用 cat 命令可以验证一下具体内容。

2.8 从远程仓库中删除文件

可以使用 cvs remove 把远程仓库中的文件标识为已删除（记得要使用 cvs commit 命令提交）。如下图：

```
[walkie@localhost 143]$ cvs remove Welcome.c
cvs remove: scheduling `Welcome.c' for removal
cvs remove: use 'cvs commit' to remove this file permanently
[walkie@localhost 143]$ cvs commit
cvs commit: Examining .
cvs commit: Examining module1
Removing Welcome.c;
/var/cvs/143/Welcome.c,v  <--  Welcome.c
new revision: delete; previous revision: 1.2
done
RCS file: /var/cvs/143/a.c,v
done
Checking in a.c;
/var/cvs/143/a.c,v  <--  a.c
initial revision: 1.1
done
[walkie@localhost 143]$
```

Shell　☐ 2 ☐ Shell

实际上这个文件并没有从仓库中被移除，它只是存储在特殊的 Attic 目录里，这样你才能还原它早先的修订版本。移除后的文件不会再传到本地沙箱。

如果你误删了某个文件，在尚未执行 cvs commit 之前，可以使用 cvs add 取消移除的操作，再以 cvs update 取回被移除的文件。

2.9 取回已经删除的文件

取回已移除的文件，其过程和取回文件的先前的修订版类似。如果你需要看一下某个文件，但是不会进行修该，可以使用：

[Walkie@localhost 143]$ cvs update -r reversion

或者

[Walkie@localhost 143]$ cvs checkout -r reversion

如果你需要修该文件，有几种方法可以让你把已移除的文件取回来进行修改。这里介绍一种，首先创建一个同名的空文件，使用 cvs add 将该文件声明为有效，然后以 cvs update –j 把空文件替换成移除操作生效之前的修订版本编号所对应的文件。如下图：

```
[walkie@localhost 143]$ touch Welcome.c
[walkie@localhost 143]$ ls
a.c  CVS  module1  Welcome.c
[walkie@localhost 143]$ cvs add Welcome.c
cvs add: Re-adding file `Welcome.c' (in place of dead revision 1.3).
cvs add: use 'cvs commit' to add this file permanently
[walkie@localhost 143]$ cvs update -j 1.2 Welcome.c
A Welcome.c
[walkie@localhost 143]$ cvs commit
cvs commit: Examining .
cvs commit: Examining module1
Checking in Welcome.c;
/var/cvs/143/Welcome.c,v  <--  Welcome.c
new revision: 1.4; previous revision: 1.3
done
[walkie@localhost 143]$ ls
a.c  CVS  module1  Welcome.c
```



2.10 移动文件或者目录

CVS 并未专门提供用来移动文件或者目录的命令。事实上，CVS 的设计并不适合进行移动操作。但是你可以组合使用各种 cvs 命令来避开这种限制。比如先 cvs remove 再使用 cvs add。

2.11 查看其他人的工程文件夹

这里推荐使用 cvs co 命令。

假设你想要看 leexcjeffrey 用户的工程目录的，则可是使用如下命令：

```
[Walkie@localhost 143]$ cvs co leexcjeffrey
```

3 CVS 的基本用法（Windows XP）

3.1 WinCVS 介绍

WinCVS 是 CVS 的一个客户端软件，它运行在 Windows 上，用来在 Windows 上登录 CVS 服务器，然后进行一些 CVS 相关的操作与管理。由于当前很多的企业内部都采用 Linux/Unix 做服务器，而用 Windows 做客户端，所以，WinCVS 与 CVS 服务器配合使用将组成最强有力的版本控制与管理的系统之一。

3.2 WinCVS 的下载与安装

最新的 WinCVS 可以从 http://sourceforge.net/project/showfiles.php?group_id=10072 地址下载到，也可以在 http://sourceforge.net/project 上下载到最新的或其它版本的 WinCVS。

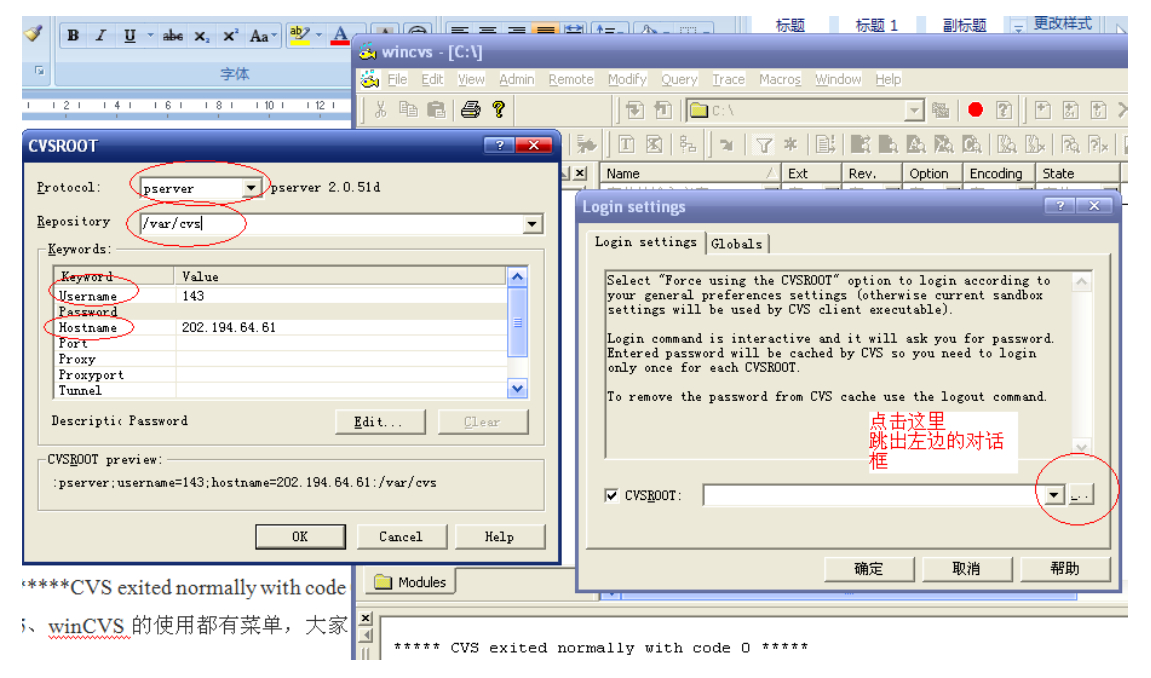
下载到相应的版本后根据向导进行安装，已经要使用 CVS 的用户，安装这个 WinCVS 应该没什么问题吧！

笔者测试使用的是最新版本 2.1.1.1。

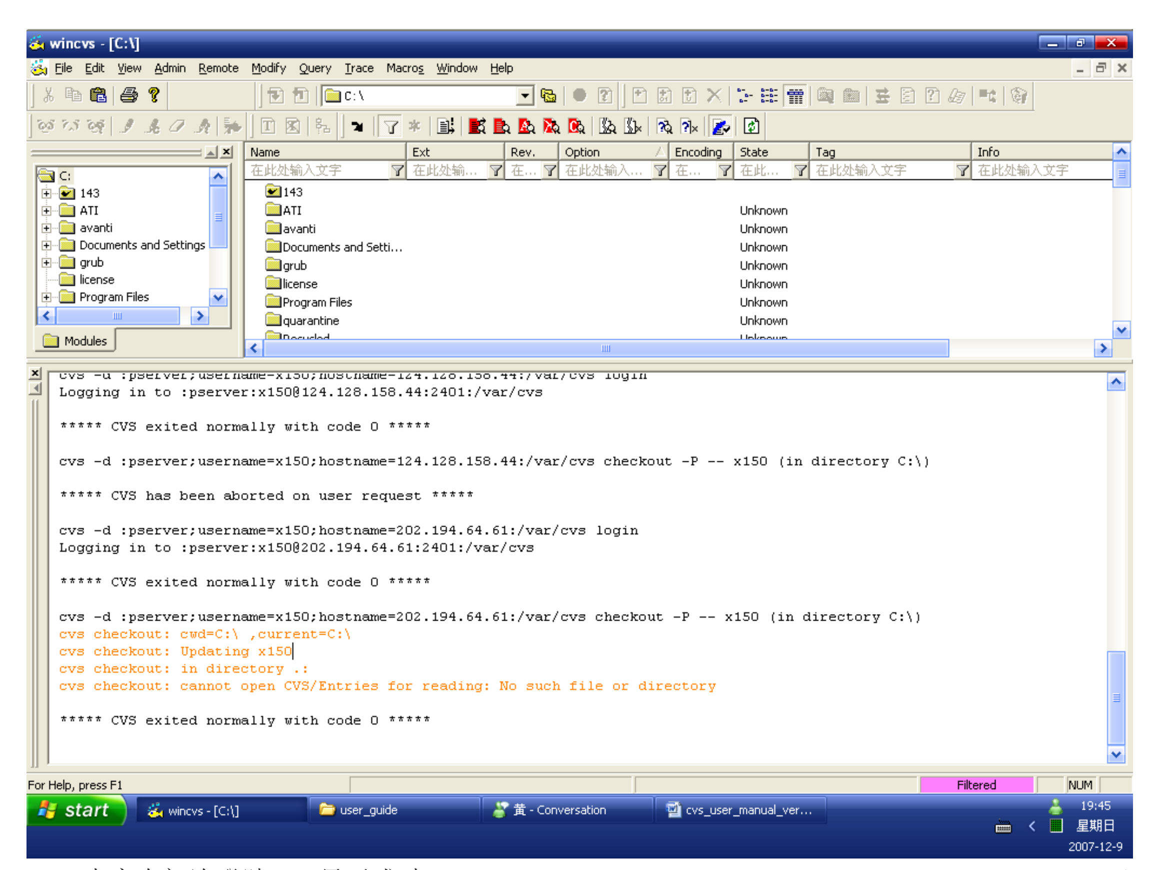
3.3 WinCVS 登录

选择 Admin->login，将出现如下对话框，点击 Login settings 对话框中红圈部分。出现配置 CVSROOT 的对话框，按图配置，输入用户名和密码，这里使用用户 143 为例。然后确认 ok 之后就会要求用户输入登录口令，wincvs 中命令的 exit code 0 表示正确执行,否则是失败，一般成功显示为

*****CVS exited normally with code 0*****



3.4 Wincvs 的初次 Checkout



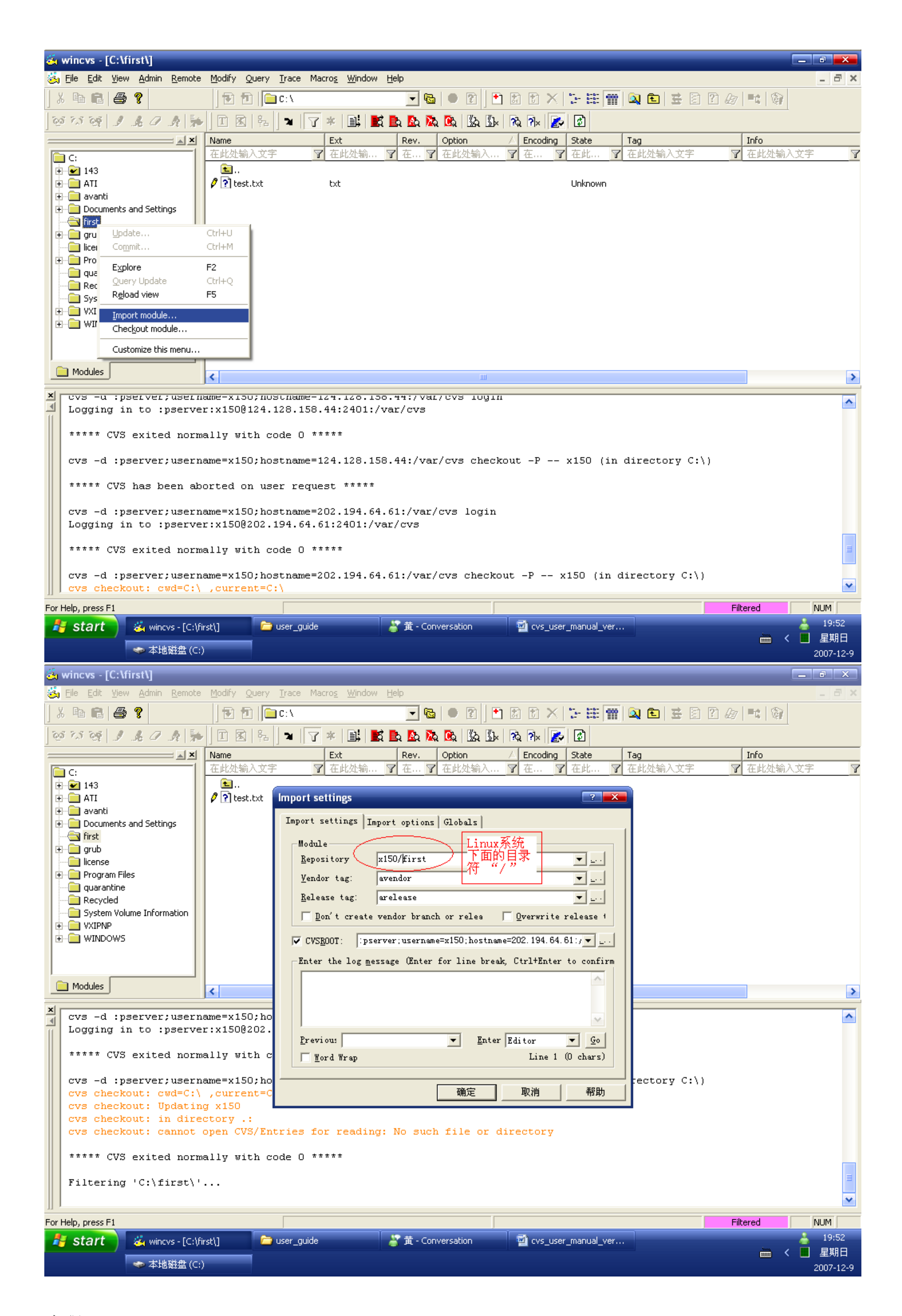
大家在初次登陆 cvs 显示成功（***** CVS exited normally with code 0 *****）之后在第一次下载大家的工程文件夹时会出现这种情况。

```
cvs -d :pserver;username=x150;hostname=202.194.64.61:/var/cvs checkout
-P -- x150 (in directory C:\)
cvs checkout: cwd=C:\ ,current=C:\
cvs checkout: Updating x150
cvs checkout: in directory .:
cvs checkout: cannot open CVS/Entries for reading: No such file or
directory

***** CVS exited normally with code 0 *****
```

然后无法正常下载你想要的下 x150 这个工程文件夹。这是因为这个文件夹现在还是空文件夹，所以在 checkout 的时候一般是无法下载下来的，所以这里的 cvs 在第一次使用的时候请参考以下使用方法。

先 import 一个随便什么文件上去，然后再把想 x150 整个工程文件夹 checkout 下载下来。如下图：



出现：

Filtering 'C:\first\'...

cvs -d :pserver;username=x150;hostname=202.194.64.61:/var/cvs import -I !

```
-I CVS -m "no message" -- x150/first avendor arelease (in directory
C:\first\)
N x150/first/test.txt

No conflicts created by this import

***** CVS exited normally with code 0 *****
```

这个时候就可以使用 3.5 小节所示的方法 checkout 出 x150 这个工程目录了

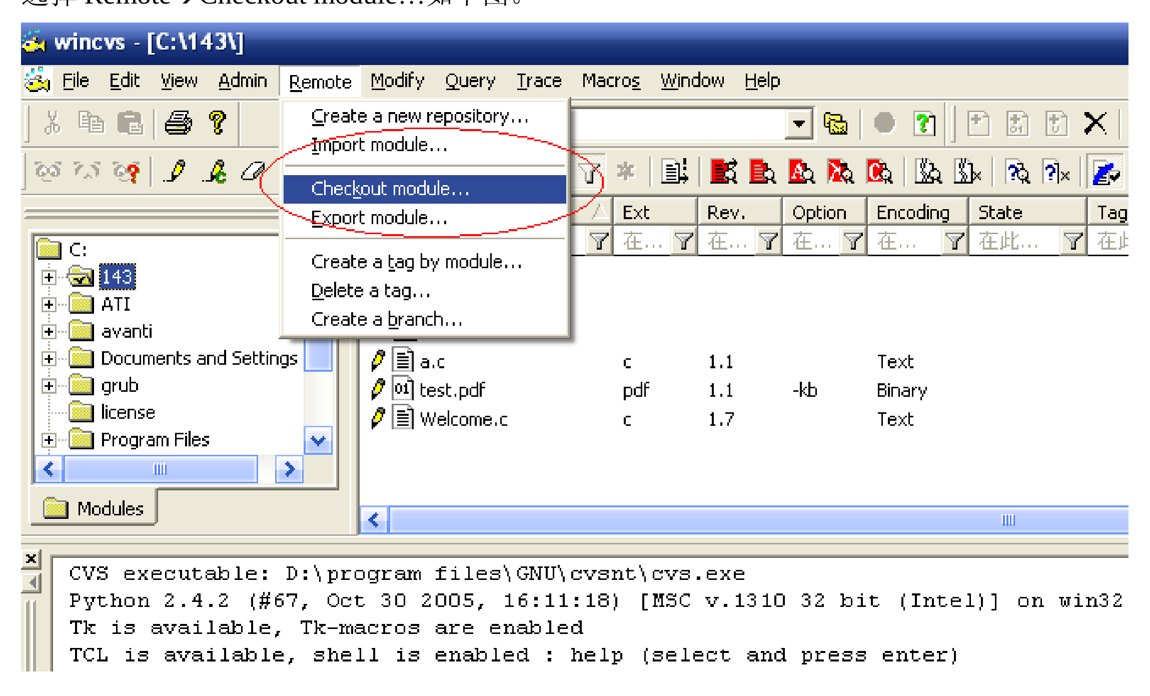
会出现如下显示内容：

```
cvs -d :pserver;username=x150;hostname=202.194.64.61:/var/cvs checkout
-P -- x150 (in directory C:\)
cvs checkout: cwd=C:\ ,current=C:\
cvs checkout: Updating x150
cvs checkout: Updating x150/first
U x150/first/test.txt

***** CVS exited normally with code 0 *****
```

3.5 下载远程仓库内容到本地沙箱中

选择 Remote→Checkout module…如下图。



会出现以下对话框，第一次选择时，需要再次设置一下 CVSROOT，如下图。

Checkout settings

Checkout settings | Update options | Merge options | Globals

Module name(s) and path on the server:

143

Check out into direc

Don't shorten module paths

Local folder to checkout to:

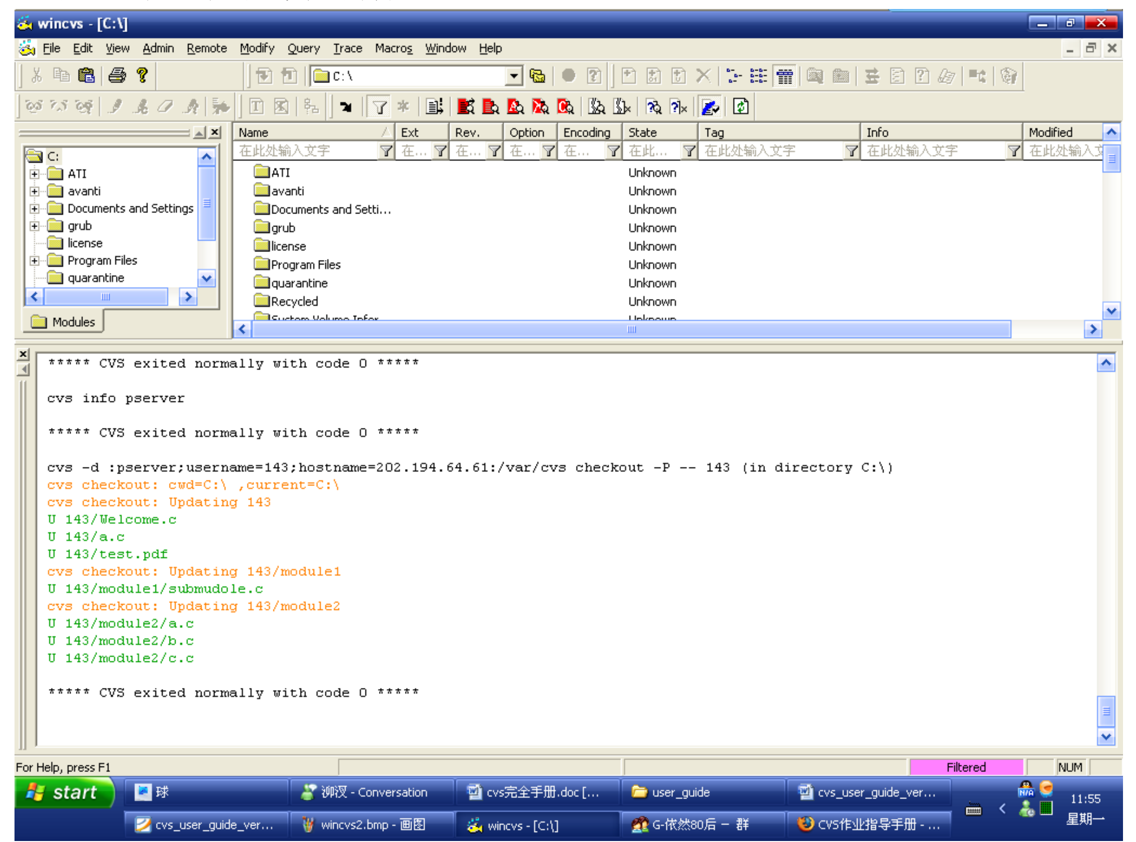
C:\

CVSROOT: :pserver;username=143;hostname=202.194.64.61:/v

Do not recurse into sub-directories

确定　取消　帮助

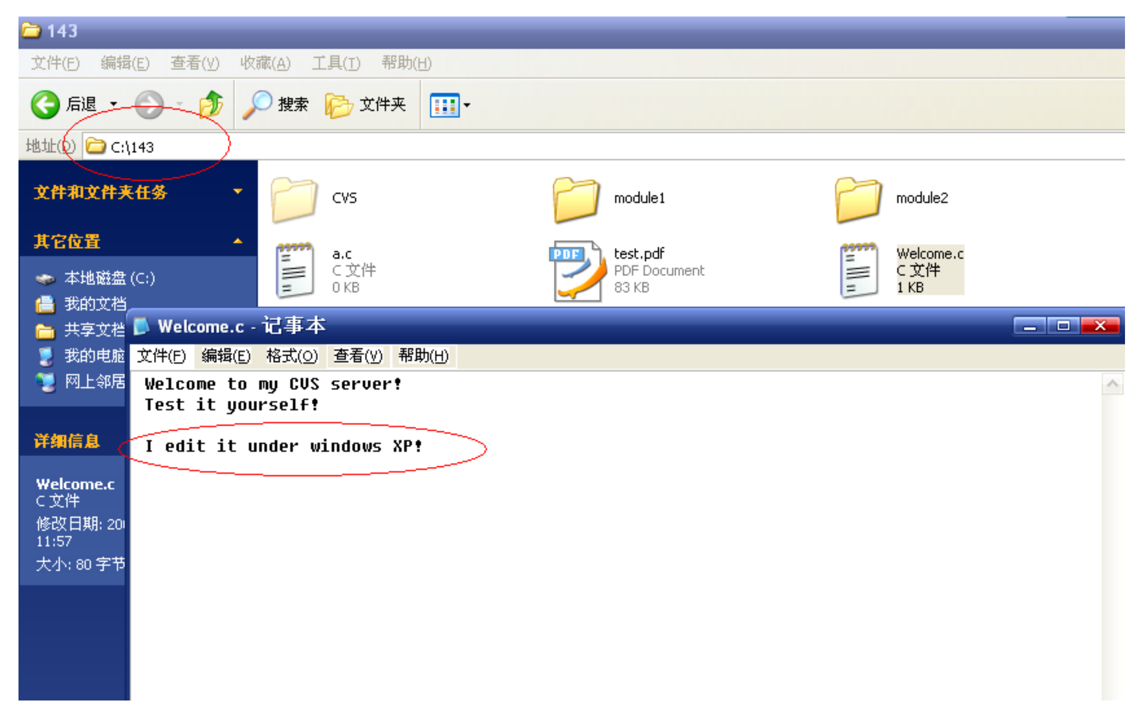
点击 OK 之后，就会出现如下界面：



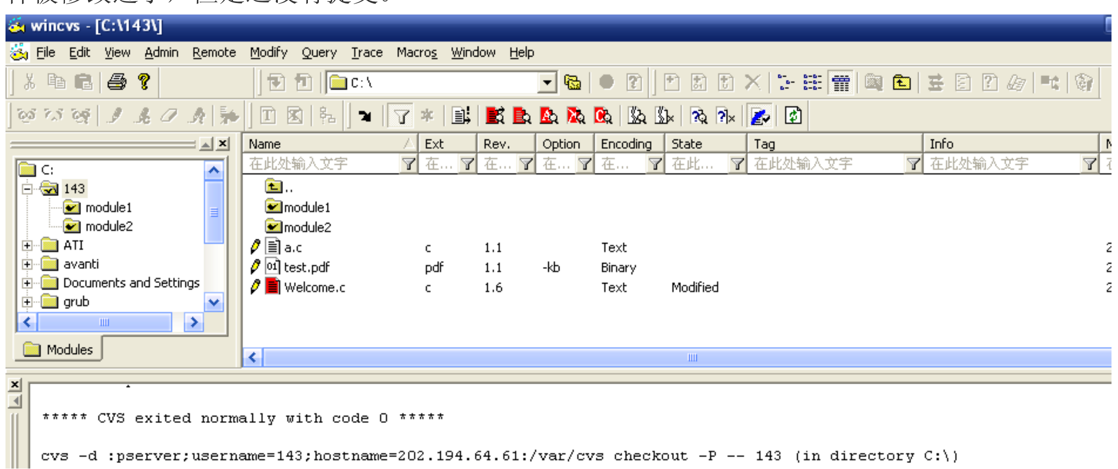
则表示下载成功。这里刷新一下左上角的 c 盘目录，143 文件夹就会出现，并且在文件夹上面会出现一个对号。表明这个文件夹是远程仓库中有的。

3.6 修改和提交

下载下来之后，就可以安装自己的习惯做修改了。例如这里我再 Welcome.c 文件里添加了一句话。



如果你这时在左上角的类似于资源管理器里，双击 143 文件夹，就会出现如下如下界面。这里的每个文件都标出了他们的状态，以及是否为非文本文件。其中红色的标志，告诉我这个文件被修改过了，但是还没有提交。

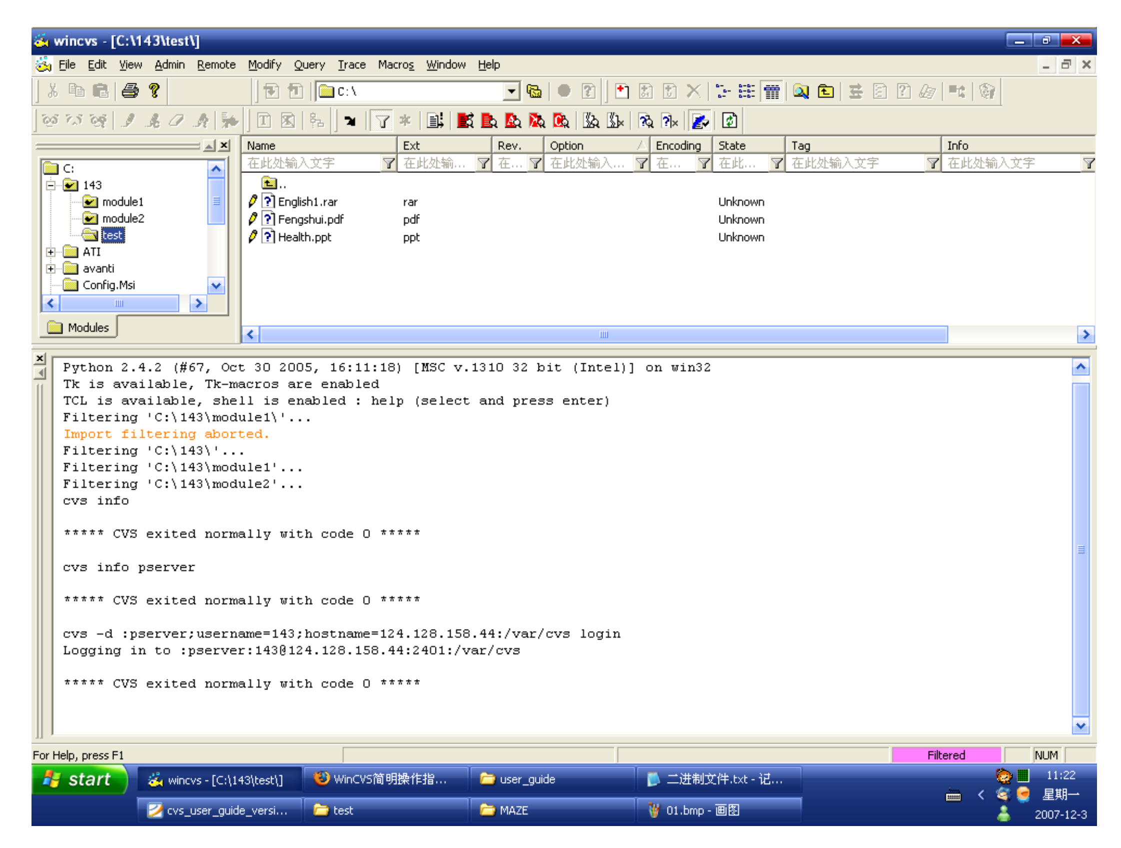


这里就可以右击这个文件，选择提交（commit）就可以了。对话框里如果出现如下信息，则表示提交成功。

```
cvs commit -m "Do it under wincvs!" -- Welcome.c (in directory C:\143\)
Checking in Welcome.c;
/var/cvs/143/Welcome.c,v  <--  Welcome.c
new revision: 1.7; previous revision: 1.6
done

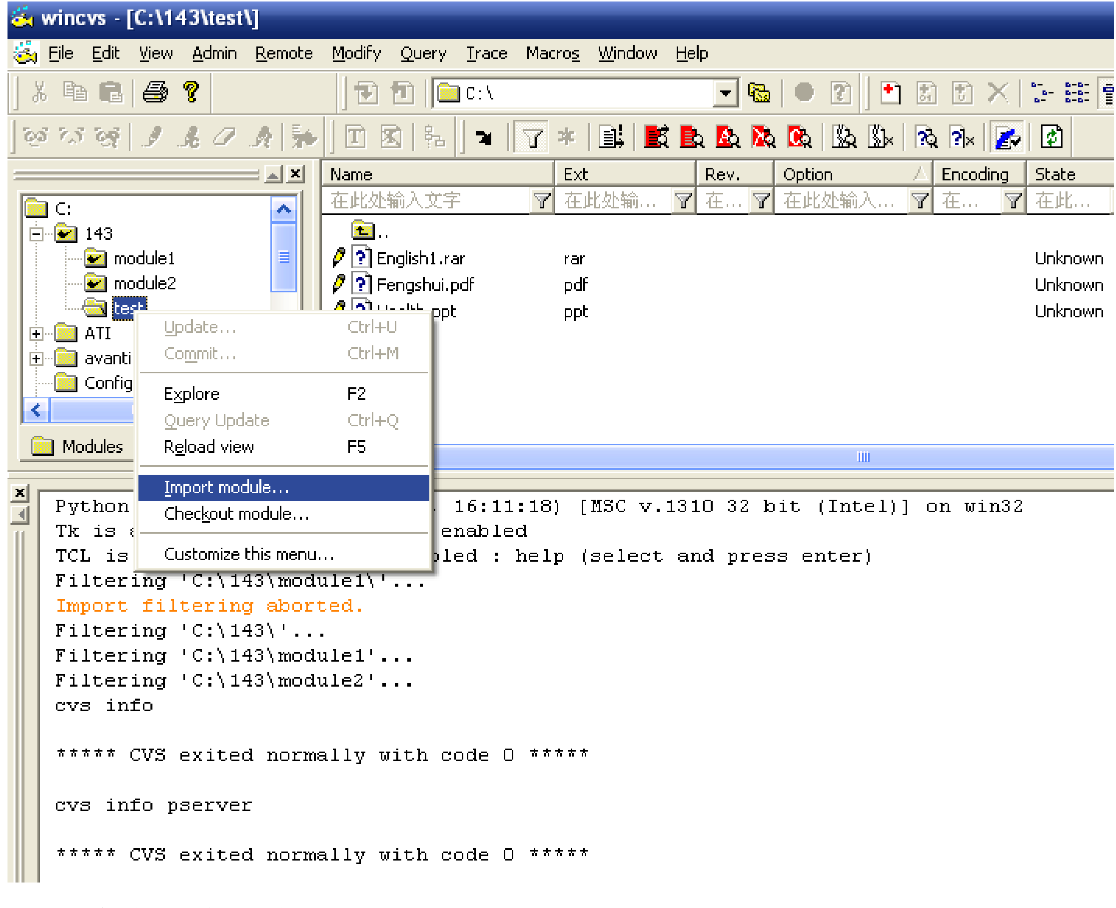
***** CVS exited normally with code 0 *****
```

3.7 上传大量文件，含非文本文件

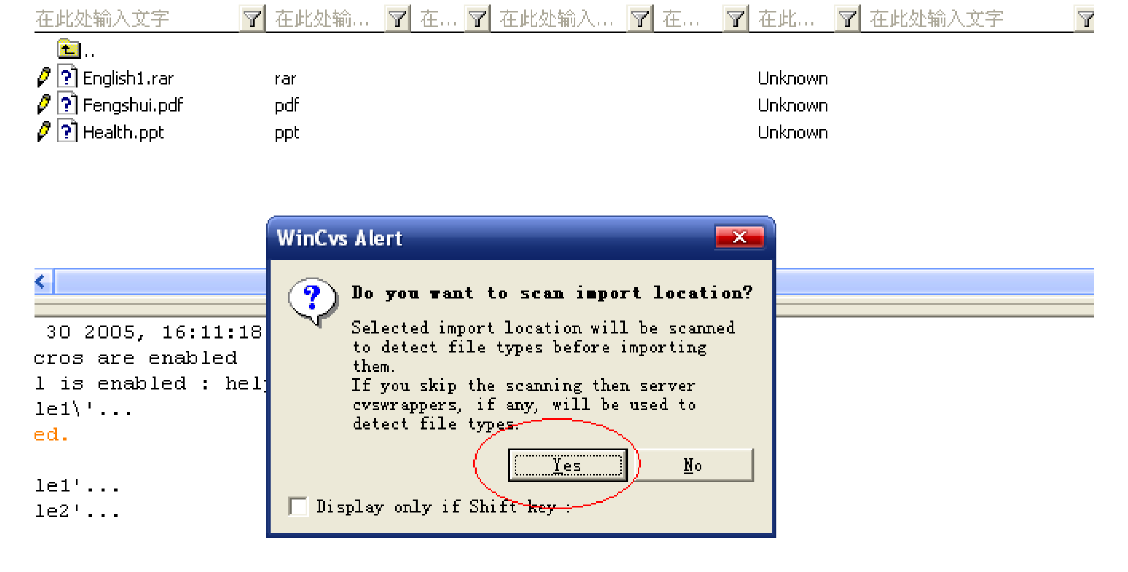


如上图，首先我们 wincvs 登录 cvs 服务器，这里还是以 143 用户为例。在本地的 143 这个本地沙箱中我们添加了 test 这个文件夹，在其中我们放置了三个非文本文件，有 word，ppt，以及 rar 文件，现在使用 import 命令上传到远程仓库中。（linux 下面如何上传见 2.3.3）

右击左边的 test 文件夹，选择 import module，如下图



就会弹出一个对话框，如下图，



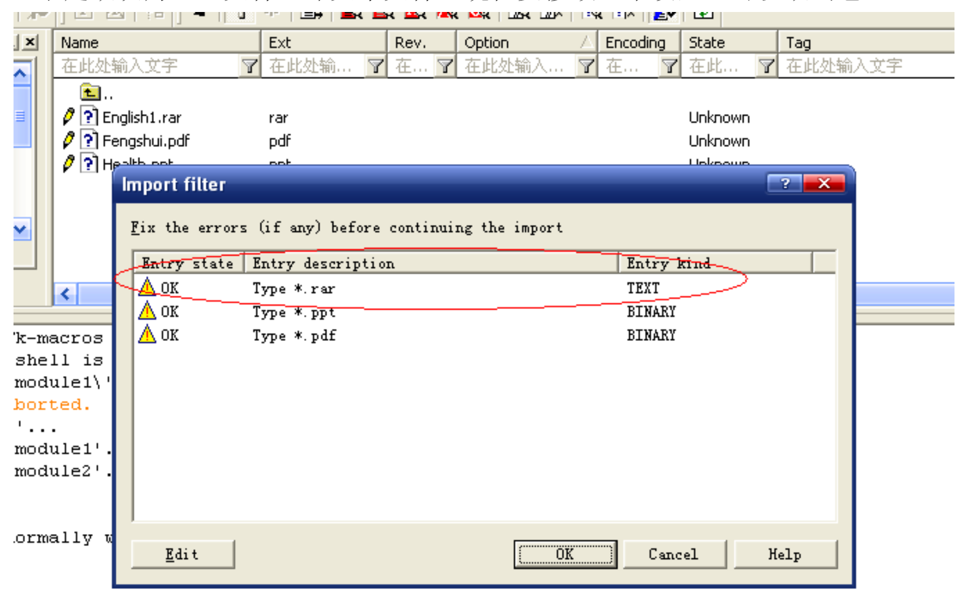
出现如下英文选项。

Selected import location will be scanned to detect file types before importing them.

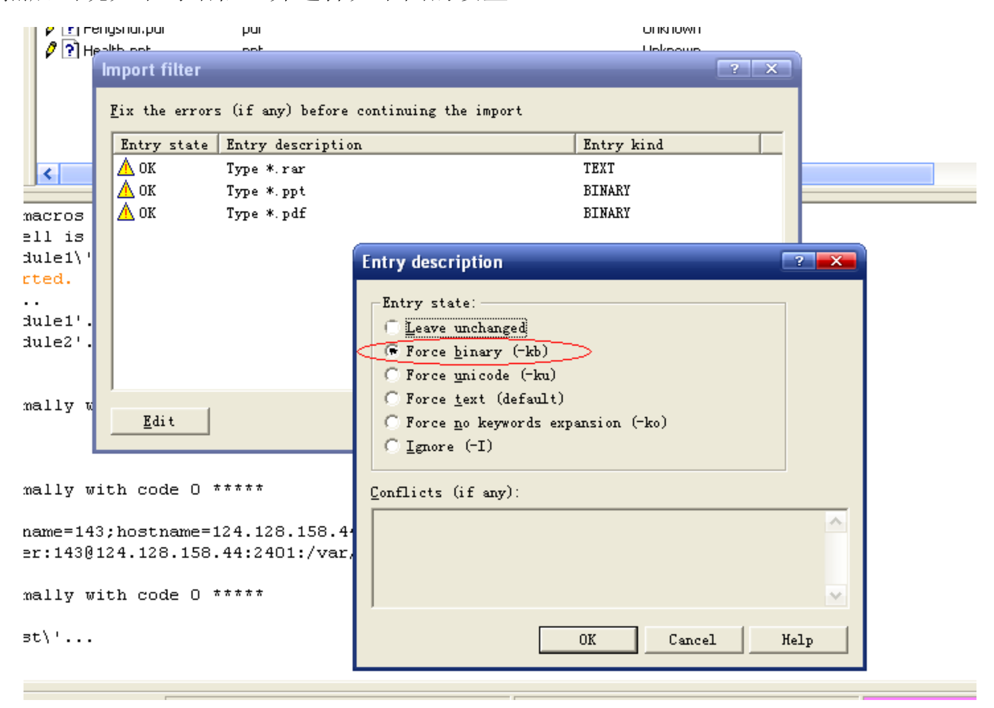
If you skip the scanning then server cvswrappers, if any, will be used to detect file types.

其实就是询问你是否要自己来设置一下文件类型，如果选择 No，那么就是用 cvs 服务器默认的设置。这里强烈建议大家选择 Yes，然后可以自己设置成非文本文件，就是二进制文件。因为在往往一个工程文件夹（ISE 或者 EDK）内部会有无数的文件，而且一般来说，windows 下面识别文件的类型是通过后缀（这也是导致 windows 下面无数病毒的一个原因），而工程文件夹内部的文件的后缀一般都很奇怪，对于你不确定的文件，建议全部设置成二进制文件。

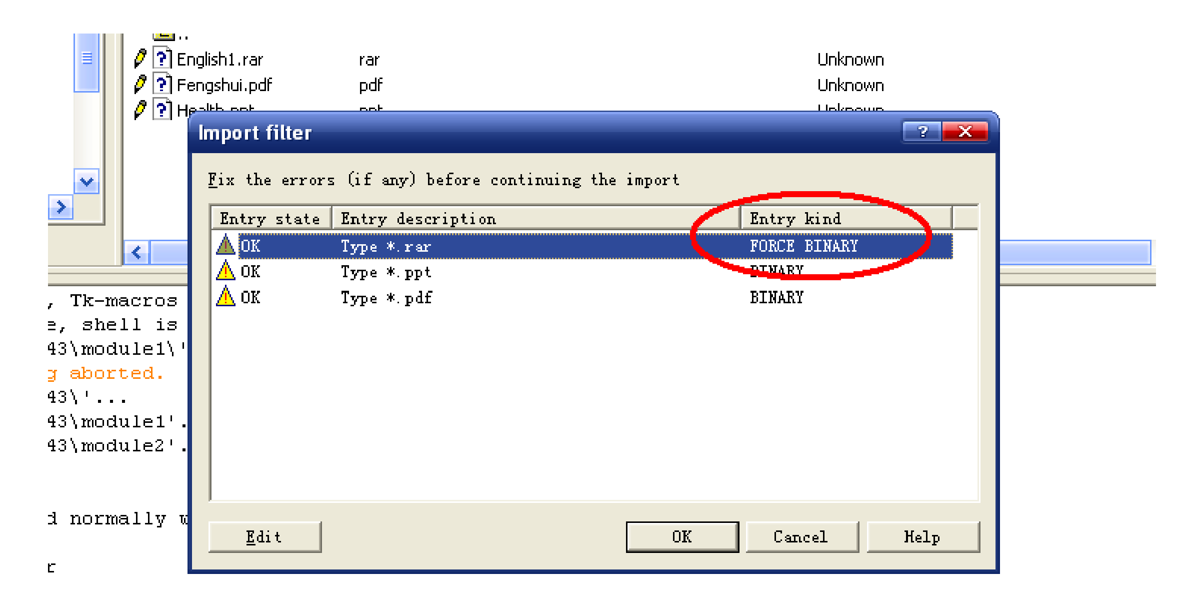
具体操作如下，上面对话框选择 Yes 之后出现如下对话框。由于之前的设置，这里已经识别出 word 和 ppt 文件不是文本文件了，于是设置成了 BINARY 文件，但是对于 rar 文件没有识别出来，于是默认为 text 文件，即文本文件。现在要修改这个设置，可以双击它。



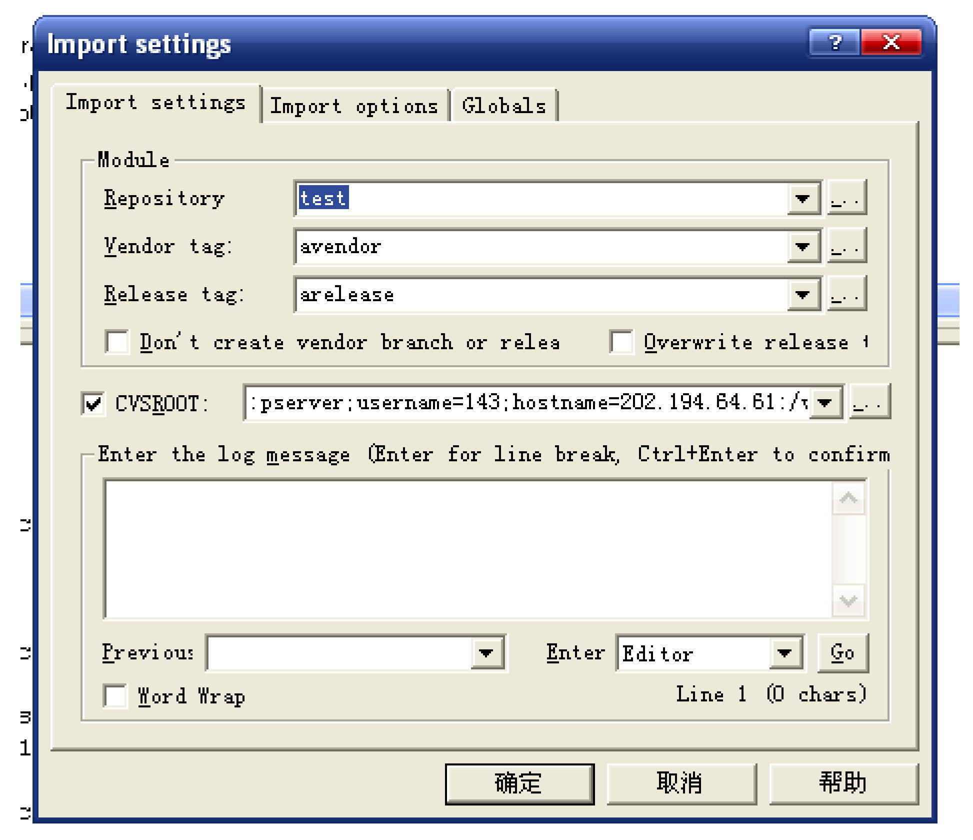
然后出现如下对话框，并选择如下图的设置



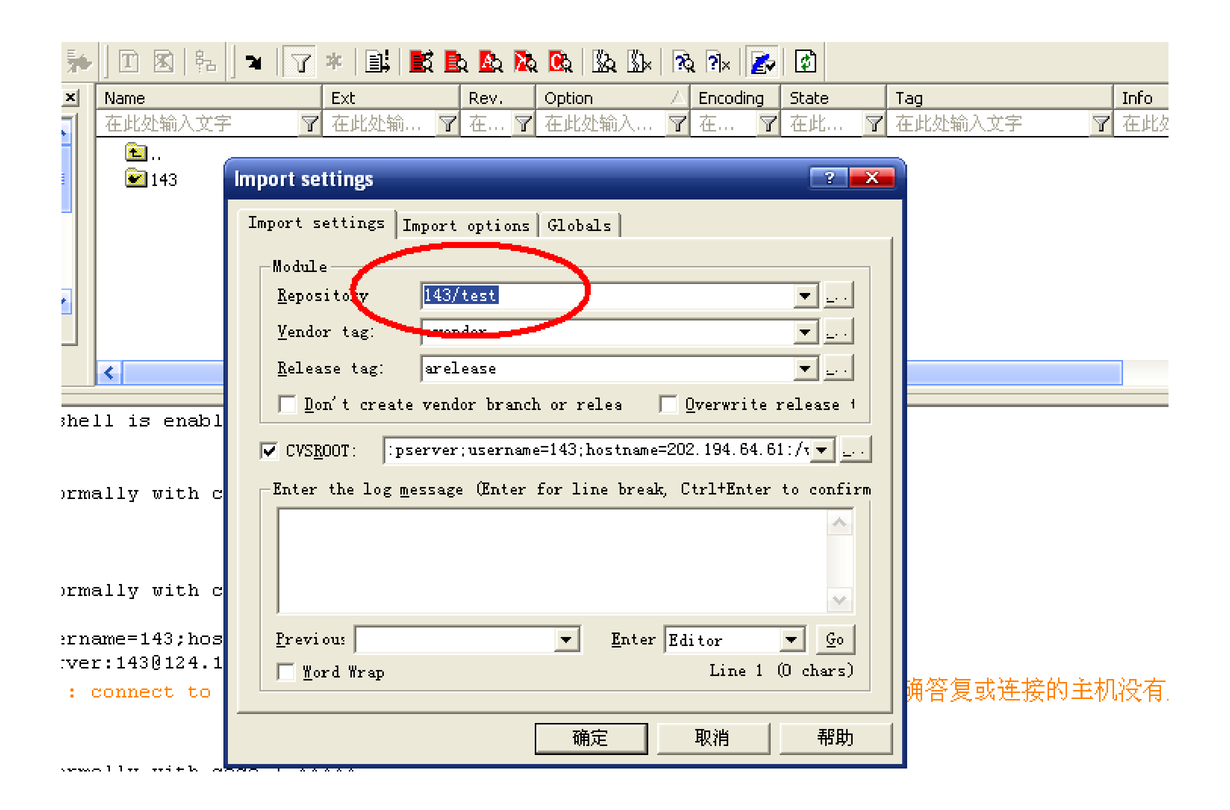
选择 OK 之后出现如下对话框，发现 rar 的文件已经强制设置成了 BINARY 文件了，这个时候就可以安确定并上传了，当然上传时还需要一些小小的设置。



点击 Ok 之后，出现如下对话框：



这里默认是把 test 这个文件夹放在仓库的根目录下面的，这是有问题的，因为大家的用户只对自己的目录有写文件的权限，（上传文件算对仓库写入文件），所以这里要设置成如下图所示：设置成自己目录下的某个文件夹或者文件。



这里还需要特别注意的一点是，这里的表示文件夹与子文件的分割分是 Linux 的“/”，不是 windows 的“\”，要不就会出现权限不够的错误。

如果成果，那么会出现以下命令行提示：

```
Filtering 'C:\143\test\'...
cvs -d :pserver;username=143;hostname=202.194.64.61:/var/cvs import -I ! -I CVS -W "*.rar -k 'b'" -W "*.ppt
"*.pdf -k 'b'" -m "no message" -- 143/test avendor arelease (in directory C:\143\test\)
N 143/test/Health.ppt
N 143/test/Fengshui.pdf
N 143/test/English1.rar

No conflicts created by this import


***** CVS exited normally with code 0 *****
```

3.8 其他

winCVS 的使用都有菜单，大家熟悉 cvs 的使用的话，用起来是很容易的，我就不多说了。比如上传文件，只要右击这个文件或者文件夹，选择 import 就可以了。但是要记住权限问题，这个文件或者文件夹一定要在你有权限修改的文件目录内部。例如这里只有放在 143 目录下的 143 用户才有权限上传到远程仓库中。

4 CVSweb（OpenHard）

CVSWEB 就是 CVS 的 WEB 界面，可以大大提高程序员定位修改的效率。

http://cvs.openhard.org

5 备注

如果在使用本文档中遇到任何问题，欢迎随时联系 Walkie（xiashengwalk@gmail.com）